IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR UNITED STATES PATENT

FOR

SYSTEM, DEVICE, AND METHOD FOR DISTRIBUTING LINK STATE INFORMATION IN A COMMUNICATION NETWORK

Inventor:

Bradley Cain
295 Harvard Street #804
Cambridge, MA 02139

Attorney Docket: 2204/184 (BA408)

Attorneys:

BROMBERG & SUNSTEIN LLP
125 Summer Street
Boston, MA 02110
(617) 443-9292

SYSTEM, DEVICE, AND METHOD FOR DISTRIBUTING LINK STATE INFORMATION IN A COMMUNICATION NETWORK

FIELD OF THE INVENTION

The present invention relates generally to communication systems, and more particularly to distributing link state information in a communication system.

BACKGROUND OF THE INVENTION

In today's information age, communication networks are often used for interconnecting computers and computer peripherals. A communication network typically includes a number of nodes that interoperate to route protocol messages. The various nodes in the communication network utilize various routing protocols in order to determine the routes that are used to route the protocol messages.

One type of routing protocol, known as a "link state" routing protocol, determines routes based upon the status of communication links between the various nodes. A link state routing protocol, such as the Open Shortest Path First (OSPF) routing protocol, requires each node to have complete topology information. Each node maintains a topology database that indicates all nodes in the communication network and lists the communication links that are associated with each node.

The various nodes in the communication network exchange link state information using link state advertisement (LSA) protocol messages. Link state information is exchanged at various times. In particular, when a node is initialized, the node needs to obtain link state information in order to determine routes for routing protocol messages, and therefore the node's neighbors send LSA protocol messages to the node in order to provide the node with the necessary link state information. Also, each node periodically tests the communication links to each of its neighbors and sends a LSA protocol message including the link status information to all of the other nodes. When a failure occurs in the communication network (such as a communication link failure or a node failure), the

various nodes in the communication network need to obtain updated link state information in order to determine new routes for routing protocol messages around the failure, and therefore the nodes adjacent to the failure send LSA protocol messages to the other nodes in the communication network in order to inform all nodes of the failure. Each node computes the routes based upon the link status information received from the other nodes.

Any time link state information is exchanged, it is desirable for the link state information to be distributed quickly in order for the receiving node(s) to determine new routes for routing protocol messages. Unfortunately, routing protocols utilize a "stop-and-wait" mechanism for distributing link state information. When a node sends an LSA protocol message to a neighbor, the node waits for an acknowledgment from the neighbor before sending another LSA protocol message. This is a very simple but inefficient way to distribute link state information.

An efficient technique for distributing link state information is needed.

SUMMARY OF THE INVENTION

In accordance with one aspect of the invention, a link state routing protocol utilizes a sliding window mechanism to efficiently distribute link state information. The sliding window mechanism permits a predetermined number of unacknowledged link state advertisement protocol messages to be outstanding at any given time. Unacknowledged link state advertisement protocol messages are retransmitted after a predetermined timeout period.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects and advantages of the invention will be appreciated more fully from the following further description thereof with reference to the accompanying drawings wherein:

FIG. 1 is a block diagram showing an exemplary communication network in accordance with an embodiment of the invention;

FIG. 2 is a block diagram depicting a series of LSA protocol messages in accordance with an embodiment of the invention;

FIG. 3 is a message flow diagram depicting an LSA protocol message exchange by a link state routing protocol using a "stop-and-wait" mechanism as known in the art;

FIG. 4 is a message flow diagram depicting an LSA protocol message exchange by a link state routing protocol using a sliding window mechanism in accordance with an embodiment of the invention;

FIG. 5 is a block diagram depicting a sliding window in a first position during an LSA protocol message exchange in accordance with an embodiment of the invention;

FIG. 6 is a block diagram depicting a sliding window in a second position during an LSA protocol message exchange in accordance with an embodiment of the invention; and

FIG. 7 is a block diagram depicting a sliding window in a third position during an LSA protocol message exchange in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

An exemplary embodiment of the present invention uses a sliding window mechanism for distributing link state information. A sliding window mechanism permits a node to send multiple protocol messages without waiting for individual acknowledgments. The node can therefore have multiple unacknowledged protocol messages outstanding at any given time. The number of unacknowledged protocol messages that are permitted to be outstanding is referred to as a "window size." A typical embodiment might use a window size of eight (8) or 128, although the window size may be set to other values and may be set based upon network characteristics (such as round trip delay to a neighbor) or other considerations. It should be noted that a window size of one (1) degenerates to a "stop-and-wait" mechanism. The neighbor may acknowledge protocol messages one at a time, or may acknowledge multiple protocol messages together. Unacknowledged protocol messages may be retransmitted after a predetermined timeout period.

In an exemplary embodiment of the present invention, a node is permitted to have a predetermined number of unacknowledged LSA protocol messages outstanding at any time. Thus, when the node needs to send link state information to a neighbor, the node is permitted to send multiple LSA protocol messages without waiting for individual acknowledgments for each LSA protocol message. The window size may be sufficient for the node to send all LSA protocol messages that it needs to send to the neighbor. However, assuming the node needs to send more LSA protocol messages than the window size allows, the node stops sending LSA protocol messages if and when the maximum number of unacknowledged outstanding LSA protocol messages is reached. The window size is preferably set so that, as long as there are no communication errors, the neighbor will acknowledge the LSA protocol messages quickly enough that the node does not have to stop sending LSA protocol messages. If a communication error occurs, the node may retransmit any unacknowledged LSA protocol messages.

There are many types of sliding window mechanisms, and an embodiment of the present invention may employ any applicable sliding window mechanism. The present invention is in no way limited to the type of sliding window mechanism employed.

In an exemplary sliding window mechanism, the node inserts a sequence number in each LSA protocol message. The sequence numbers typically start at zero (0) and increment to the window size minus one (1) before wrapping back to zero (0). In other words, the sequence number is modulo N, where N is the window size. The neighbor may be required to acknowledge each individual LSA protocol message, for example, by sending the sequence number of the LSA protocol message being acknowledged, or the neighbor may be permitted to acknowledge multiple consecutive LSA protocol messages, for example, by sending the sequence number of the last LSA protocol message in the sequence of LSA protocol messages being acknowledged. The neighbor may or may not be permitted to acknowledge LSA protocol messages received out of sequence or received after a dropped or missing LSA protocol message. Various retransmission techniques require the node to save unacknowledged LSA protocol messages in case the node needs to retransmit one or more LSA protocol messages. The node may be required to retransmit

all LSA protocol messages following an unacknowledged LSA protocol message, or may be permitted to retransmit individual unacknowledged LSA protocol messages.

FIG. 1 is a block diagram showing an exemplary communication network 100 in accordance with an embodiment of the invention. The communication network 100 includes, among other things, a node 102 and a neighbor 106 coupled over a communication link 104. The node 102 and the neighbor 106 interoperate to route protocol messages within the communication network 100. Specifically, the node 102 and the neighbor 106 implement a link state routing protocol for determining routes within the communication network 100.

Implementation of the link state routing protocol requires the node 102 and the neighbor 106 to exchange LSA protocol messages over the communication link 104. During a particular LSA protocol message exchange, the node 102 may need to send multiple LSA protocol messages to the neighbor 106 over the communication link 104. FIG. 2 depicts a series of eleven (11) protocol messages 200 to be sent by the node 102 to the neighbor 106 over the communication link 104.

In a prior art embodiment, the node 102 sends the series of LSA protocol messages 200 to the neighbor 106 over the communication link 104 using a "stop-and-wait" mechanism. FIG. 3 is a message flow diagram depicting an LSA protocol message exchange by a link state routing protocol using a "stop-and-wait" mechanism. The node 102 sends one LSA protocol message at a time to the neighbor 106, and waits for an acknowledgment from the neighbor 106 before sending another LSA protocol message to the neighbor 106. Thus, the node 102 sends the first LSA protocol message 302 to the neighbor 106, and waits for an acknowledgment from the neighbor 106. Upon receiving the acknowledgment 304 from the neighbor 106, the node 102 sends the second LSA protocol message 306 to the neighbor 106, and waits for an acknowledgment from the neighbor 106. Upon receiving the acknowledgment 308 from the neighbor 106, the node 102 proceeds to send subsequent LSA protocol messages (not shown) in the same manner, until all LSA protocol messages have been sent and acknowledged.

In an exemplary embodiment of the invention, the node 102 sends the series of LSA protocol messages 200 to the neighbor 106 over the communication link 104 using a

sliding window mechanism. FIG. 4 is a message flow diagram depicting an LSA protocol message exchange by a link state routing protocol using a sliding window mechanism using a window size of eight (8). The node 102 is permitted to have up to eight (8) unacknowledged LSA protocol messages outstanding at any given time. Thus, the node 102 may send the first eight (8) LSA protocol messages (402, 404, 406, 408, 410, 412, 414, 416) to the neighbor 106 before the node 102 must stop to wait for an acknowledgment from the neighbor 106 for at least one of the LSA protocol messages. At this point, there are eight (8) unacknowledged LSA protocol messages outstanding, specifically the LSA protocol messages (402, 404, 406, 408, 410, 412, 414, 416). FIG. 5 depicts the position of the sliding window 502 after the node 102 sends the first eight (8) LSA protocol messages to the neighbor 106.

Upon receiving an acknowledgment 418 for the first LSA protocol message 402 from the neighbor 106, the node 102 sends the ninth LSA protocol message 420 to the neighbor 106. At this point, there are eight (8) unacknowledged LSA protocol messages outstanding, specifically the LSA protocol messages (404, 406, 408, 410, 412, 414, 416, 420). FIG. 6 depicts the position of the sliding window 602 after the node 102 sends the ninth LSA protocol message 420 to the neighbor 106.

Upon receiving an acknowledgment 422 for the second and third LSA protocol messages (404, 406) from the neighbor 106, the node 102 sends the tenth and eleventh LSA protocol messages (424, 426) to the neighbor 106. At this point, there are eight (8) unacknowledged LSA protocol messages outstanding, specifically the LSA protocol messages (408, 410, 412, 414, 416, 420, 424, 426). FIG. 7 depicts the position of the sliding window 702 after the node 102 sends the tenth and eleventh LSA protocol messages (424, 426) to the neighbor 106.

In actuality, the neighbor 106 typically sends acknowledgments as it receives and processes the LSA protocol messages. The neighbor 106 may acknowledge each LSA protocol message individually, or, as with the acknowledgment 422 shown in FIG. 4, the neighbor 106 may acknowledge multiple LSA protocol messages simultaneously. Thus, the node 102 typically receives one or more acknowledgment messages from the neighbor 106 before reaching the maximum number of outstanding unacknowledged LSA protocol

messages, and therefore the node 102 generally does not have to stop sending LSA protocol messages in order to wait for acknowledgment messages from the neighbor 106.

In the course of the LSA protocol message exchange between the node 102 and the neighbor 106, it is possible for certain protocol messages to be lost. This may be due to a communication error on the communication link 104 or other reason. Protocol messages may be lost in either direction. For example, the neighbor 106 may not receive one or more LSA protocol messages sent by the node 102, in which case the neighbor does not acknowledge the lost LSA protocol message(s). Also, the node 102 may not receive one or more acknowledgment messages sent by the neighbor 106. In either case, the node 102 detects the error condition by failing to receive an acknowledgment message for a particular LSA protocol message within a predetermined timeout period. The node 102 preferably retransmits unacknowledged LSA protocol messages after a predetermined timeout period. The node 102 may retransmit only specific unacknowledged LSA protocol messages, although it is more typical for the node 102 to retransmit all LSA protocol messages from the unacknowledged LSA protocol message onward. Thus, assuming the LSA protocol messages (408, 410, 412, 414, 416, 420, 424, 426) are outstanding, as depicted by the sliding window 702 as shown in FIG. 7, and the LSA protocol message 408 is unacknowledged, then the node 102 typically retransmits the LSA protocol messages (408, 410, 412, 414, 416, 420, 424, 426). This is true even if the neighbor 106 received some or all of the LSA protocol messages after the LSA protocol message 408.

In various embodiments of the invention, the sliding window mechanism may be integrated with an existing link state routing protocol, such as OSPF, or may be used in a new link state routing protocol. The present invention is in no way limited to a specific link state routing protocol.

In an exemplary embodiment of the present invention, predominantly all of the routing protocol logic including the sliding window logic is implemented as a set of computer program instructions that are stored in a computer readable medium and executed by an embedded microprocessor system within a node. Various embodiments of the invention may be implemented in any conventional computer programming language. For example, an embodiment may be implemented in a procedural programming language

(*e.g.*, "C") or an object oriented programming language (*e.g.*, "C++"). Alternative embodiments of the invention may be implemented using discrete components, integrated circuitry, programmable logic used in conjunction with a programmable logic device such as a Field Programmable Gate Array (FPGA) or microprocessor, or any other means including any combination thereof.

Alternative embodiments of the invention may be implemented as a computer program product for use with a computer system. Such implementation may include a series of computer instructions fixed either on a tangible medium, such as a computer readable media (*e.g.*, a diskette, CD-ROM, ROM, or fixed disk), or fixed in a computer data signal embodied in a carrier wave that is transmittable to a computer system via a modem or other interface device, such as a communications adapter connected to a network over a medium. The medium may be either a tangible medium (*e.g.*, optical or analog communications lines) or a medium implemented with wireless techniques (*e.g.*, microwave, infrared or other transmission techniques). The series of computer instructions embodies all or part of the functionality previously described herein with respect to the system. Those skilled in the art should appreciate that such computer instructions can be written in a number of programming languages for use with many computer architectures or operating systems. Furthermore, such instructions may be stored in any memory device, such as semiconductor, magnetic, optical or other memory devices, and may be transmitted using any communications technology, such as optical, infrared, microwave, or other transmission technologies. It is expected that such a computer program product may be distributed as a removable medium with accompanying printed or electronic documentation (*e.g.*, shrink wrapped software), preloaded with a computer system (*e.g.*, on system ROM or fixed disk), or distributed from a server or electronic bulletin board over the network (*e.g.*, the Internet or World Wide Web).

The present invention may be embodied in other specific forms without departing from the essence or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive.